TRABAJO DE GRADO
Opción Seminario-Diplomado.

Proyectos Crehana Seminario De Machine Learning

Corporación Universitaria Remington

Facultad de Ingenierías

Ingeniería de Sistemas

Estudiante: Juan Pablo Serna Quintero

Tutor: Juan Pablo Vélez Uribe

Opción de Trabajo de grado Seminario

2024

TRABAJO DE GRADO
Opción Seminario-Diplomado.

TRABAJO DE GRADO
Opción Seminario-Diplomado.

Tabla de contenido

UNIREMINGTON
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

TRABAJO DE GRADO
Opción Seminario-Diplomado.

Proyecto 1: Análisis confractual

Machine Learning - Analisis contractual.Rmd

C: > Users > juanp > Desktop > U > proyectos finales seminario > Proyecto final - analisis confractual > Machine Learning - Analisis contractual.Rmd

```
---
title: 'Proyecto Machine Learning: Análisis contrafactual'
author: "Juan Pablo Serna"
date: "2024-02-14"
output: html_document
---

## AVANCE 1: Diseño experimental y asignación aleatoria

# Cargar datos de clientes interesados
clientes_interesados <- read.csv("clientes_interesados.csv")

# Segmentación de clientes en tres grupos
set.seed(123)  # Fijar semilla para reproducibilidad
clientes_interesados$grupo <- sample(c("A", "B", "C"),
                                     size = nrow(clientes_interesados),
                                     replace = TRUE)

# Agregar columna con identificador de grupo
write.csv(clientes_interesados, "clientes_interesados_con_grupo.csv", row.names = FALSE)

## AVANCE 2: Evaluación de impacto

# Cargar datos de ventas
ventas <- read.csv("datos_ventas.csv")

# Unir datos de ventas con información de grupos de promoción
ventas_con_grupos <- merge(ventas, clientes_interesados, by = "id_cliente")

# Analizar impacto de las promociones en las ventas
library(dplyr)

impacto_promociones <- ventas_con_grupos %>%
  group_by(grupo) %>%
  summarise(ventas_totales = sum(ventas_totales),
            clientes_totales = n_distinct(id_cliente))

print(impacto_promociones)

## AVANCE 3: Focalización de la estrategia

# Cargar datos de clientes con información de promoción
clientes_promocion <- read.csv("clientes_promocion.csv")

# Seleccionar variables relevantes
variables <- c("edad", "sexo", "ingreso", "grupo_promocion", "compra")

```

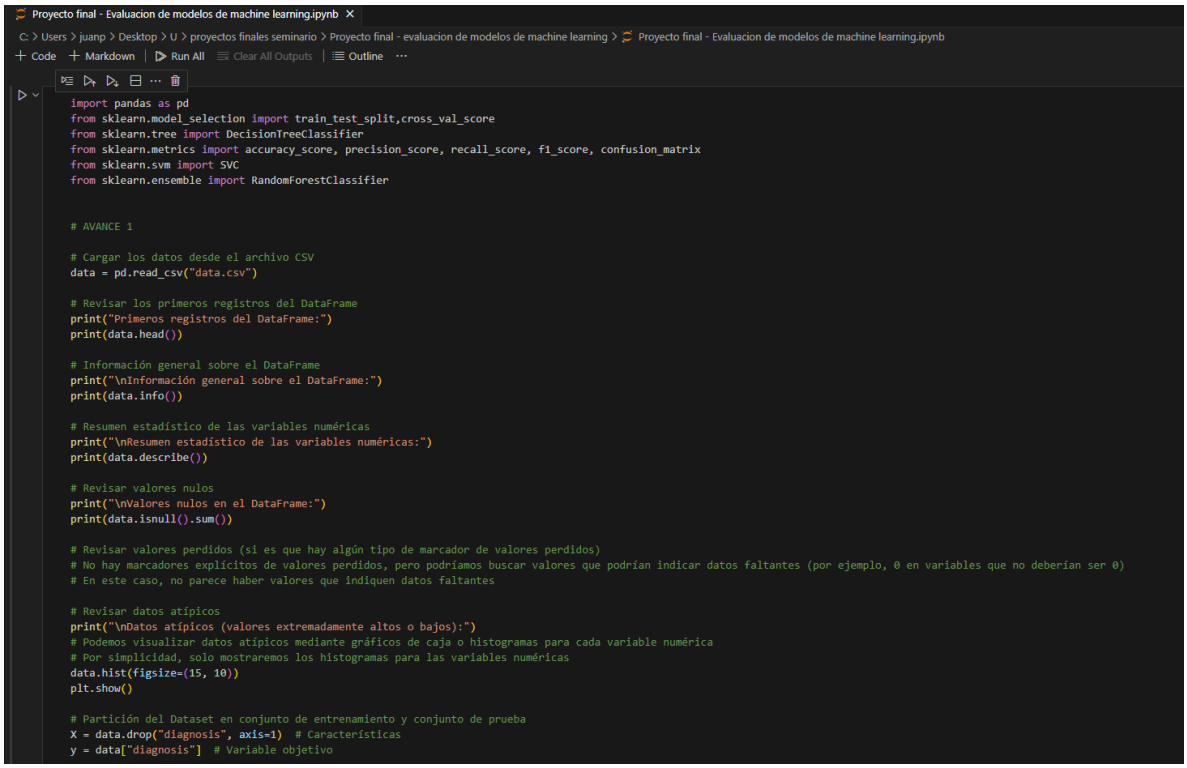
TRABAJO DE GRADO
Opción Seminario-Diplomado.

```
47
48  datos_modelo <- clientes_promocion[, variables]
49
50  # Dividir los datos en conjunto de entrenamiento y prueba
51  set.seed(123)  # Fijar semilla para reproducibilidad
52  indices_entrenamiento <- sample(1:nrow(datos_modelo), 0.8 * nrow(datos_modelo))
53  datos_entrenamiento <- datos_modelo[indices_entrenamiento, ]
54  datos_prueba <- datos_modelo[-indices_entrenamiento, ]
55
56  # Instalar y cargar el paquete 'grf' para Causal Forest
57  install.packages("grf")
58  library(grf)
59
60  # Entrenar el modelo causal
61  modelo_causal <- causal_forest(grupo_promocion ~ ., data = datos_entrenamiento)
62
63  # Predecir el efecto causal sobre los datos de prueba
64  efecto_causal <- predict(modelo_causal, X = datos_prueba)
65
66  # Calcular el efecto medio del tratamiento
67  efecto_medio_tratamiento <- mean(efecto_causal$predictions)
68
69  print(paste("El efecto medio del tratamiento es:", efecto_medio_tratamiento))
70
```

UNIREMINGTON®
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

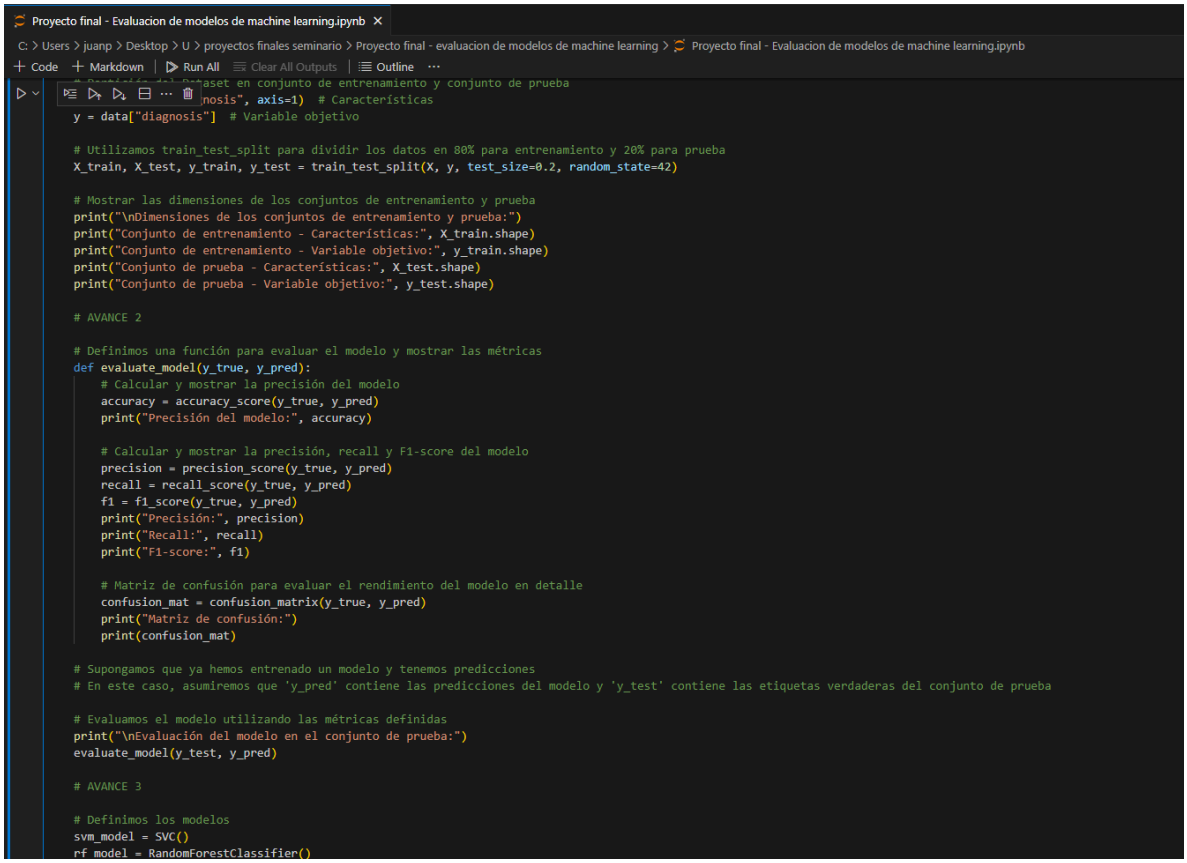
TRABAJO DE GRADO
Opción Seminario-Diplomado.

Proyecto 2: Evaluación de modelos de machine learning



UNIREMINGTON®
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

TRABAJO DE GRADO
Opción Seminario-Diplomado.



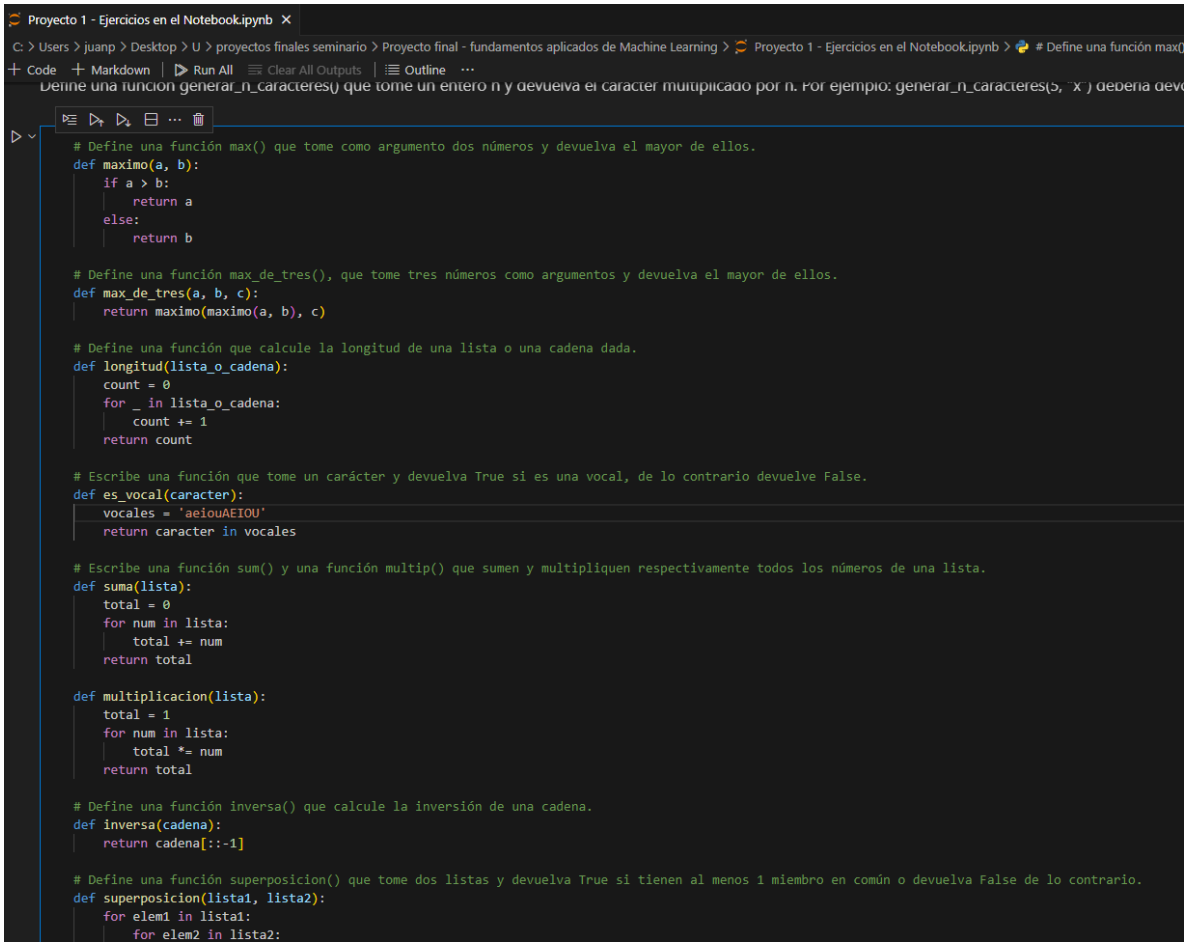

```
# Validación cruzada para evaluar el rendimiento de SVM
svm_scores = cross_val_score(svm_model, X_train, y_train, cv=5, scoring='accuracy')
print("Puntuaciones de validación cruzada para SVM:", svm_scores)
print("Precisión media de SVM:", svm_scores.mean())

# Validación cruzada para evaluar el rendimiento de Random Forest
rf_scores = cross_val_score(rf_model, X_train, y_train, cv=5, scoring='accuracy')
print("\nPuntuaciones de validación cruzada para Random Forest:", rf_scores)
print("Precisión media de Random Forest:", rf_scores.mean())

# Comparación de modelos y selección del mejor
if svm_scores.mean() > rf_scores.mean():
    print("\nSVM tiene mejor precisión media. Seleccionando SVM como el modelo final.")
    final_model = svm_model
else:
    print("\nRandom Forest tiene mejor precisión media. Seleccionando Random Forest como el modelo final.")
    final_model = rf_model}
```

UNIREMINGTON®
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

TRABAJO DE GRADO
Opción Seminario-Diplomado.

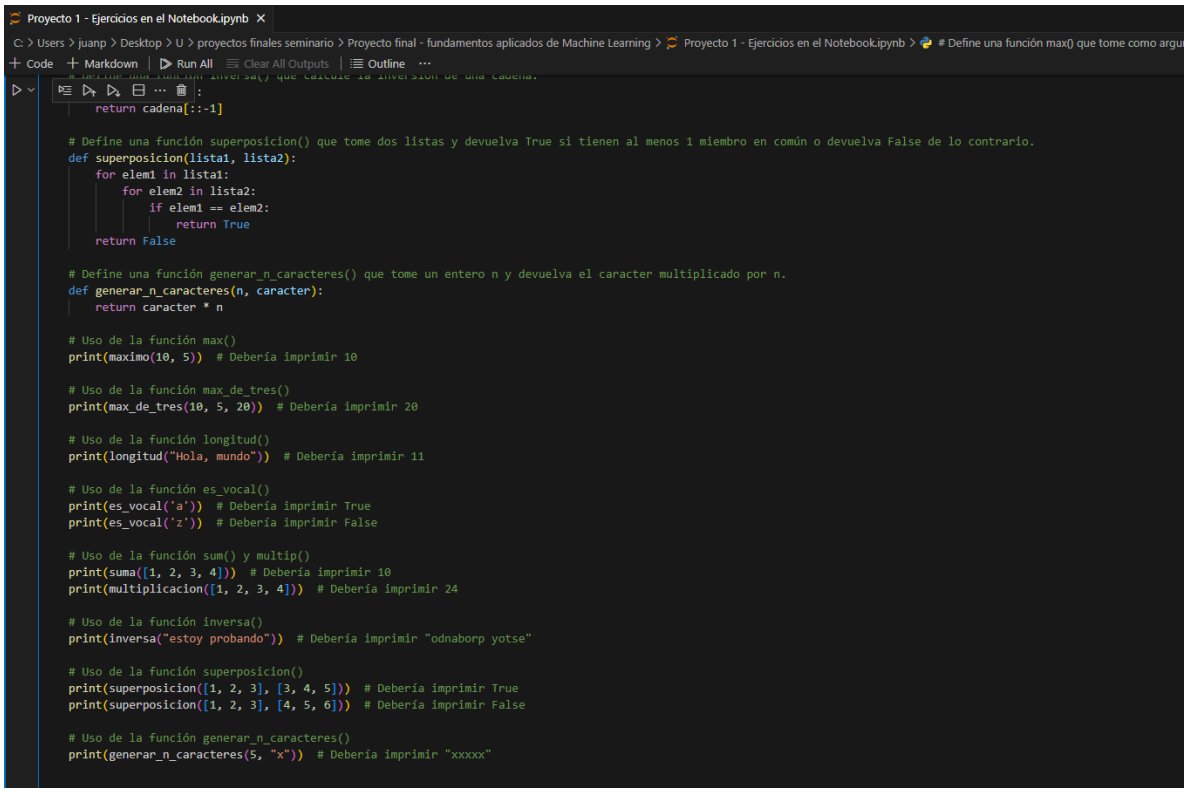
Proyecto 3: Fundamentos aplicados de machine learning

Ejercicio 1:



UNIREMINGTON®
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

TRABAJO DE GRADO
Opción Seminario-Diplomado.



Ejercicio 2:

```
# define "X" como todo el conjunto de variables explicativas
# define "y" como la variable a predecir (precios)

X =  df.drop('variable_objetivo', axis=1)
y =  df['variable_objetivo']

# usa "train_test_split" adicionando los parámetros test_size = 0.2 y random_state = 42

X_train, X_test, y_train, y_test = train_test_split( X, y, test_size=0.2, random_state=42 ) # 4 argumentos


# crea una instancia para la regresión lineal
reg_all = LinearRegression()

# usa el método "fit" de la regresión con los datos de entrenamiento (tanto para X como para y)
reg_all.fit( X_train, y_train )

# usa el método "predict" de la regresión para hacer las predicciones sobre el conjunto X de entrenamiento
y_train_predict = reg_all.predict( X_train )

# calcula el rmse
# pasa los parámetros adecuados a "mean_squared_error". Recuerda que esta métrica se calcula
# usando la diferencia entre
rmse = (np.sqrt(mean_squared_error(y_train, y_train_predict)))

# calcula el R cuadrado
r2 = round(reg_all.score(X_train, y_train),2)

print("Desempeño del modelo para los datos de entrenamiento")
print("--------------------------------------")
print('RMSE: {}'.format(rmse))
print('R2: {}'.format(r2))
print("\n")
```

TRABAJO DE GRADO
Opción Seminario-Diplomado.

```
# repite el proceso para los datos de test

y_pred = reg_all.predict(X_test)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
r2 = round(reg_all.score(X_test, y_test), 2)

print("Desempeño del modelo para los datos de test")
print("--------------------------------------")
print("RMSE: {}".format(rmse))
print("R2: {}".format(r2))
print("\n")
```

- Compara frente al modelo guiado durante el módulo que construiste usando una única variable. ¿Cómo cambian los pronósticos en test y entrenamiento?

Esperamos ver predicciones mejoradas en los datos de prueba al pasar de un modelo de regresión lineal simple a uno más complejo que utiliza todas las variables disponibles.

```
Visualiza tus predicciones:
```

```
# pasa los precios reales de test y los pronósticos para graficarlos
plt.scatter(y_test, y_pred) # dos argumentos

# estas 5 líneas de código no necesitas modificarlas
plt.xlabel("Precio Real ($1000)")
plt.ylabel("Precio Pronosticado ($1000)")
plt.xticks(range(0, int(max(y_test)),2))
plt.yticks(range(0, int(max(y_test)),2))
plt.title("Precios Reales vs. Precios Pronosticados de Casas")
```

Otras preguntas:

- Para qué predecir? Para Tomar decisiones informadas. Los pronósticos pueden proporcionar información valiosa para la toma de decisiones en una variedad de contextos, desde estrategias comerciales hasta políticas públicas.
- Investiga qué empresas utilizan pronóstico de precios Empresas minoristas: utilice la previsión de precios para optimizar las estrategias de precios, gestionar el inventario y planificar promociones.

Empresas de servicios financieros: Utilizan previsiones de precios para predecir movimientos en los mercados financieros, evaluar riesgos y tomar decisiones de inversión.

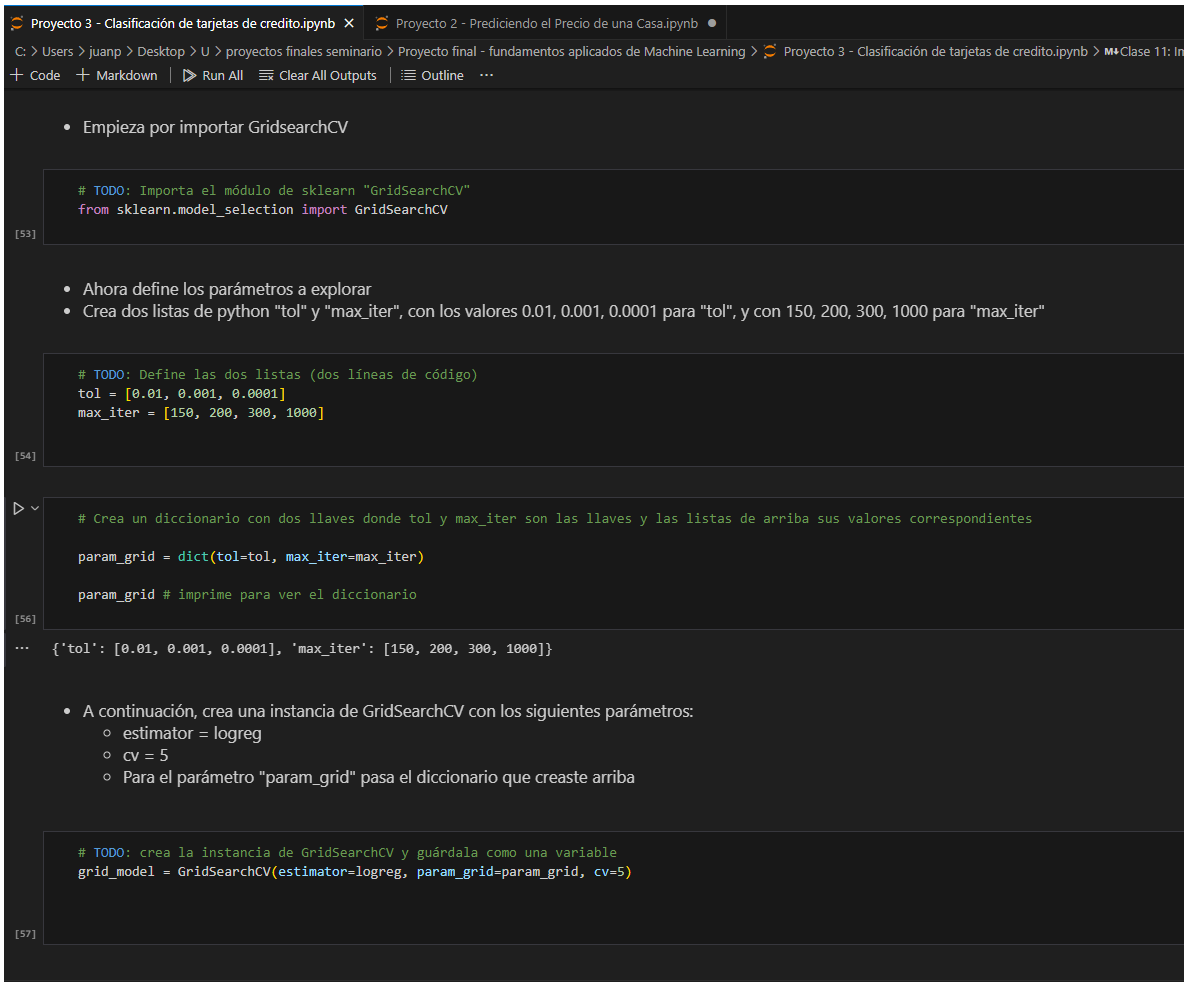
Empresas inmobiliarias: utilice pronósticos de precios para predecir las tendencias del mercado inmobiliario, valorar propiedades y tomar decisiones de inversión.

Empresas de transporte y logística: utilice previsiones de precios para planificar rutas, gestionar la demanda y optimizar la utilización de recursos.

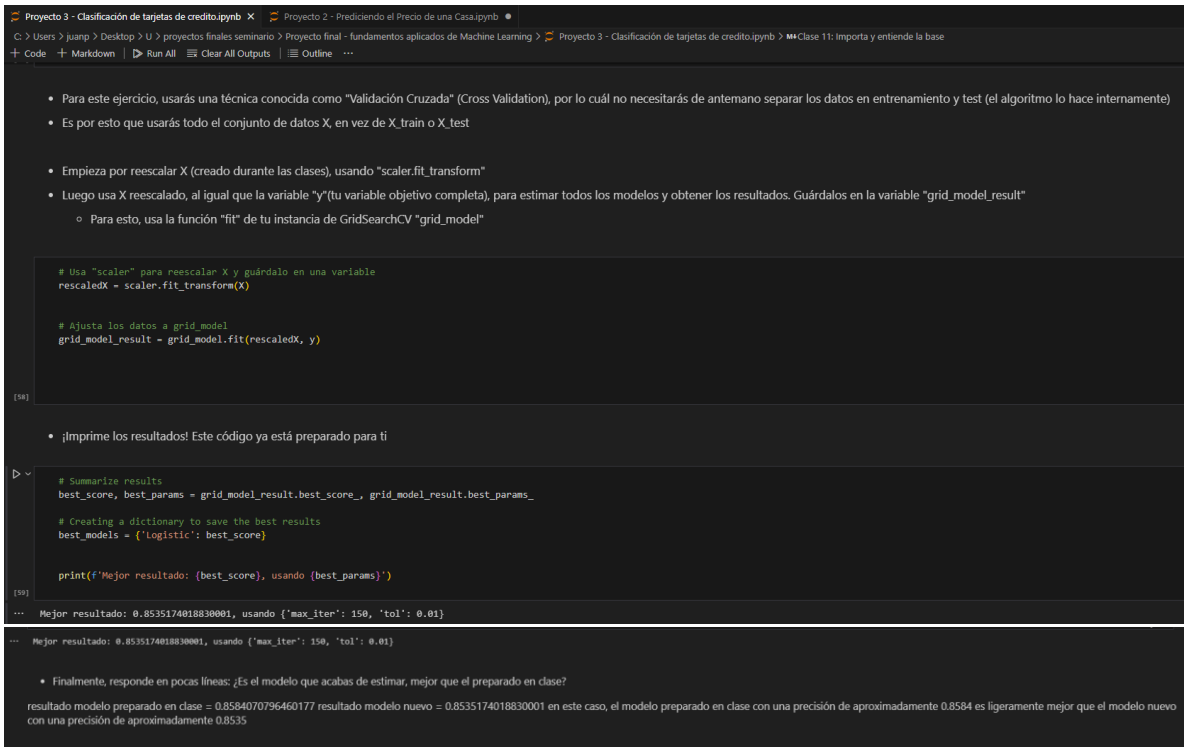
Empresas de tecnología: utilice la previsión de precios para fijar los precios de productos y servicios y predecir la demanda de productos tecnológicos.

- Aplicabilidad de este modelo de Boston hoy? Si bien el modelo de Boston sigue siendo una herramienta valiosa para examinar las relaciones entre las características y los precios de la vivienda, su aplicabilidad directa a los mercados inmobiliarios modernos puede requerir consideración adicional y potencial.

Ejercicio 3:


Proyecto 3 - Clasificación de tarjetas de credito.ipynb
Proyecto 2 - Prediciendo el Precio de una Casa.ipynb
C: > Users > juanp > Desktop > U > proyectos finales seminario > Proyecto final - fundamentos aplicados de Machine Learning > Proyecto 3 - Clasificación de tarjetas de credito.ipynb
Code
Markdown
Run All
Clear All Outputs
Outline
Empieza por importar GridsearchCV
# TODO: Importa el módulo de sklearn "GridSearchCV"
from sklearn.model_selection import GridSearchCV
[53]
Ahora define los parámetros a explorar
Crea dos listas de python "tol" y "max_iter", con los valores 0.01, 0.001, 0.0001 para "tol", y con 150, 200, 300, 1000 para "max_iter"
# TODO: Define las dos listas (dos líneas de código)
tol = [0.01, 0.001, 0.0001]
max_iter = [150, 200, 300, 1000]
[54]
# Crea un diccionario con dos llaves donde tol y max_iter son las llaves y las listas de arriba sus valores correspondientes
param_grid = dict(tol=tol, max_iter=max_iter)
param_grid # imprime para ver el diccionario
[56]
{'tol': [0.01, 0.001, 0.0001], 'max_iter': [150, 200, 300, 1000]}
A continuación, crea una instancia de GridSearchCV con los siguientes parámetros:
estimator = logreg
cv = 5
Para el parámetro "param_grid" pasa el diccionario que creaste arriba
# TODO: crea la instancia de GridSearchCV y guárdala como una variable
grid_model = GridSearchCV(estimator=logreg, param_grid=param_grid, cv=5)
[57]

TRABAJO DE GRADO
Opción Seminario-Diplomado.



Proyecto 4: Innovación tecnológica con inteligencia artificial

Base de datos elegida

Top 50 canciones del 2023

Preguntas de investigación.

- ¿Cuáles son las características musicales más comunes en las canciones exitosas del 2023?
- ¿Existe alguna correlación entre el género de la canción y su popularidad?
- ¿Qué características influyen más en la duración de una canción en el top 50?
- ¿Hay alguna tendencia temporal en las características musicales de las canciones exitosas?
- ¿Se puede predecir el éxito de una canción utilizando características como el tempo, la energía y la valencia?

TRABAJO DE GRADO
Opción Seminario-Diplomado.

Pregunta seleccionada y columnas parametrizadas.

¿Se puede predecir el éxito de una canción utilizando características como el tempo, la energía y la valencia?

- tempo: El tempo de la canción en BPM.
- energy: La energía percibida de la canción.
- valence: La positividad de la canción.
- popularity: La popularidad de la canción en una escala de 0 a 100 (nuestro objetivo de predicción).

Pasos a seguir y elección de IA

Un modelo de regresión lineal o un modelo de bosque aleatorio pueden ser adecuados para este propósito, dada la naturaleza constante de la popularidad de las canciones.

import pandas as pd Untitled-3 ● # Importar librerías necesarias Untitled-1 ●

```python
# Importar librerías necesarias
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score

# Cargar la base de datos
data = pd.read_csv("top_50_2023.csv")

# Seleccionar características y objetivo
X = data[['tempo', 'energy', 'valence']]
y = data['popularity']

# Dividir los datos en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Inicializar y entrenar el modelo de bosque aleatorio
model = RandomForestRegressor(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Predecir la popularidad en el conjunto de prueba
y_pred = model.predict(X_test)

# Evaluar el rendimiento del modelo
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print("Mean Squared Error:", mse)
print("R^2 Score:", r2)
```

Representación gráfica de los resultados

```
# Predecir la popularidad en el conjunto de prueba
y_pred = model.predict(X_test)

# Evaluar el rendimiento del modelo
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print("Mean Squared Error:", mse)
print("R^2 Score:", r2)

# Graficar predicciones vs valores reales
plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred, color='blue', alpha=0.5)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'k--', lw=2)
plt.xlabel('Popularidad Real')
plt.ylabel('Popularidad Predicha')
plt.title('Predicciones vs Valores Reales')
plt.grid(True)
plt.show()
```

Proyecto 5: Introducción a la ética en la inteligencia artificial

a. ¿Debería la compañía invertir en corregir el sesgo, incluso si eso significa un retraso en la implementación y posiblemente afectar su posición en el mercado?

R/ La corrección de la distorsión en el algoritmo es de importancia crucial desde una perspectiva ética y responsabilidad social. Aunque la corrección sería un retraso en la implementación y, a corto plazo, podría influir en su posición en el mercado, una corrección ética fortalecerá la reputación de la compañía a largo plazo y su compromiso con la equidad en la atención médica.

b. Si la compañía decide corregir el sesgo, es posible que mucha gente no logre obtener un diagnóstico de manera veloz o precisa, ¿vale la pena corregir el sesgo incluso si afecta negativamente a las personas que no se veían afectadas anteriormente?

R/ Aunque la corrección puede conducir a una reducción temporal en la velocidad o precisión del diagnóstico para ciertos grupos, no para perpetuar la discriminación y el riesgo de diagnósticos incorrectos para los grupos afectados. Es muy importante determinar la prioridad en la justicia y la precisión en el diagnóstico de efectividad comercial a corto plazo.

c. ¿Cómo debería la compañía abordar las acusaciones de infracción de violación de la privacidad? ¿Deberían compensar a los pacientes cuyos datos se utilizaron sin permiso, y cómo podría hacerse esto sin crear un precedente peligroso?

R/ La Compañía debe tratar las acusaciones de violación de la protección de datos con transparencia y responsabilidad. Con respecto a la compensación, la compañía podría considerar ofrecer a los pacientes afectados servicios médicos gratuitos o descuentos en tratamientos futuros para remediar el daño causado.

d. ¿Hasta qué punto debería la compañía revelar cómo funciona su algoritmo?

R/ La transparencia en el funcionamiento del algoritmo es necesaria para aumentar la confianza entre los trabajadores médicos y entre los pacientes. Aunque la compañía puede proteger ciertos aspectos de su propiedad intelectual, debe dar una descripción general de cómo se capacita el algoritmo, qué datos se usan y cómo se lleva a cabo la precisión.

e. ¿Cómo pueden los líderes de la compañía equilibrar las demandas de todas estas áreas y asegurar que su tecnología no solo sea comercialmente exitosa sino también éticamente responsable?

R/ Los líderes de la compañía deben adoptar un enfoque proactivo para superar los problemas éticos y de privacidad relacionados con su tecnología. Al hacerlo, la compañía puede mantener su competitividad en el mercado mientras cumple con sus obligaciones éticas y sociales.

UNIREMINGTON
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

TRABAJO DE GRADO
Opción Seminario-Diplomado.

Proyecto 6: Introducción a la inteligencia artificial

Ejercicio 1:

```python
from keras.models import load_model  # TensorFlow is required for Keras to work
from PIL import Image, ImageOps  # Install pillow instead of PIL
import numpy as np

# Disable scientific notation for clarity
np.set_printoptions(suppress=True)

# Load the model
model = load_model("keras_Model.h5", compile=False)

# Load the labels
class_names = open("labels.txt", "r").readlines()

# Create the array of the right shape to feed into the keras model
# The 'length' or number of images you can put into the array is
# determined by the first position in the shape tuple, in this case 1
data = np.ndarray(shape=(1, 224, 224, 3), dtype=np.float32)

# Replace this with the path to your image
image = Image.open("<IMAGE_PATH>").convert("RGB")

# resizing the image to be at least 224x224 and then cropping from the center
size = (224, 224)
image = ImageOps.fit(image, size, Image.Resampling.LANCZOS)

# turn the image into a numpy array
image_array = np.asarray(image)

# Normalize the image
normalized_image_array = (image_array.astype(np.float32) / 127.5) - 1

# Load the image into the array
data[0] = normalized_image_array

# Predicts the model
prediction = model.predict(data)
index = np.argmax(prediction)
class_name = class_names[index]
confidence_score = prediction[0][index]

# Print prediction and confidence score
print("Class:", class_name[2:], end="")
print("Confidence Score:", confidence_score)
```

Ejercicio 2:


Plantilla_NN_CIFAR100.ipynb
C: > Users > juanp > Desktop > U > proyectos finales seminario > Proyecto final - introduccion a la inteligencia artificial > Plantilla_NN_CIFAR100.ipynb > empty cell
Code
Markdown
Run All
Clear All Outputs
Outline
Importando librerías
import tensorflow as tf
from tensorflow import keras
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Flatten, Dense
[18]
Carga de Datos
Importamos las librerías que utilziaremos para cargar los datos para la red neuronal:
from tensorflow.keras.datasets import cifar100
[10]
Corre la siguiente linea para cargar tu dataset
(train_images, train_labels), (test_images, test_labels) = cifar100.load_data()
[14]
Revisa las dimensiones de tus datos
print("Dimensiones de los datos de entrenamiento:")
print("Imágenes de entrenamiento:", train_images.shape)
print("Etiquetas de entrenamiento:", train_labels.shape)
print("\nDimensiones de los datos de prueba:")
print("Imágenes de prueba:", test_images.shape)
print("Etiquetas de prueba:", test_labels.shape)
[15]

UNIREMINGTON
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

TRABAJO DE GRADO
Opción Seminario-Diplomado.

Plantilla_NN_CIFAR100.ipynb

C: > Users > juanp > Desktop > U > proyectos finales seminario > Proyecto final - introduccion a la inteligencia artificial > Plantilla_NN_CIFAR100.ipynb > empty cell

+ Code + Markdown | Run All | Clear All Outputs | Outline ···

```
Dimensiones de los datos de entrenamiento:
Imágenes de entrenamiento: (50000, 32, 32, 3)
Etiquetas de entrenamiento: (50000, 1)

Dimensiones de los datos de prueba:
Imágenes de prueba: (10000, 32, 32, 3)
Etiquetas de prueba: (10000, 1)
```

Abre el primer registro de train

```
print("Primer registro del conjunto de datos de entrenamiento:")
print("Etiqueta del primer registro:", train_labels[0])
```

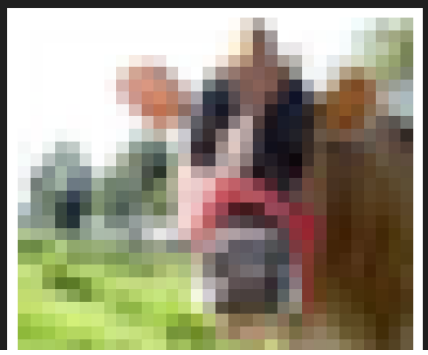
[16]

```
Primer registro del conjunto de datos de entrenamiento:
Etiqueta del primer registro: [19]
```

Visualiza la primer imagen del dataset

```
plt.imshow(train_images[0])
plt.axis('off')  # Para eliminar los ejes
plt.show()
```

[17]



Modelo

Crea tu red neuronal con una forma de 32*32 px

```
[19]
model = Sequential([
    Flatten(input_shape=(32, 32, 3)),
    Dense(128, activation='relu'),
    Dense(100, activation='softmax')
])

model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 flatten (Flatten)           (None, 3072)              0

 dense (Dense)               (None, 128)               393344

 dense_1 (Dense)             (None, 100)               12900

=================================================================
Total params: 406244 (1.55 MB)
Trainable params: 406244 (1.55 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

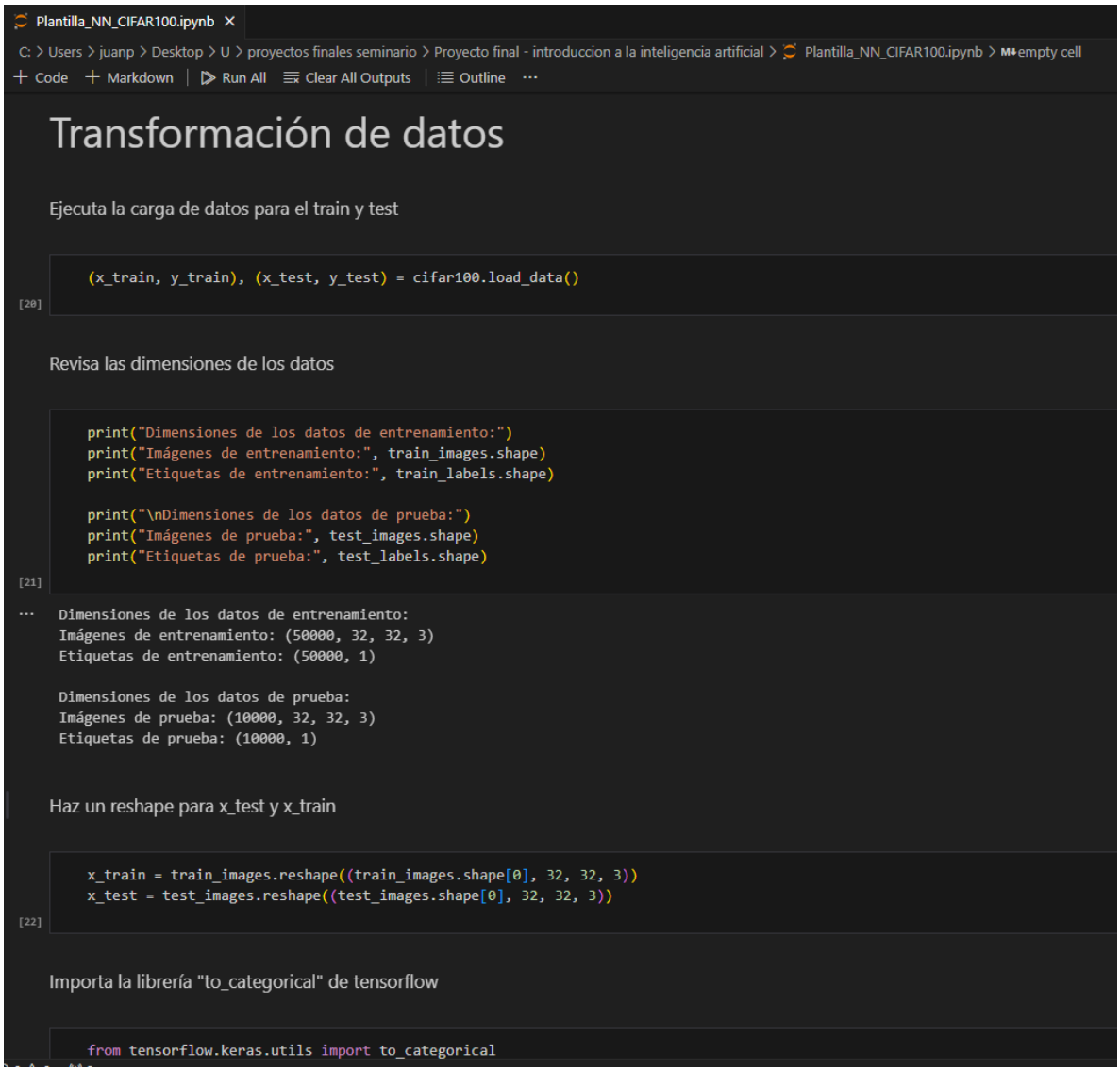
Transformación de datos

Ejecuta la carga de datos para el train y test

```
[20]
(x_train, y_train), (x_test, y_test) = cifar100.load_data()
```

UNIREMINGTON
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

TRABAJO DE GRADO
Opción Seminario-Diplomado.



TRABAJO DE GRADO
Opción Seminario-Diplomado.

Plantilla_NN_CIFAR100.ipynb

C: > Users > juanp > Desktop > U > proyectos finales seminario > Proyecto final - introduccion a la inteligencia artificial > Plantilla_NN_CIFAR100.i

+ Code + Markdown | Run All | Clear All Outputs | Outline ···

Convierte a categoricas las variables de y_train y y_test

```
[24]
y_train_categorical = to_categorical(y_train)
y_test_categorical = to_categorical(y_test)
```

Imprime el primer registro de las etiquetas de train Considera que es un vector [x,x]

```
[25]
print("Primer registro de las etiquetas de entrenamiento:")
print(y_train_categorical[0])
```

```
Primer registro de las etiquetas de entrenamiento:
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0.]
```

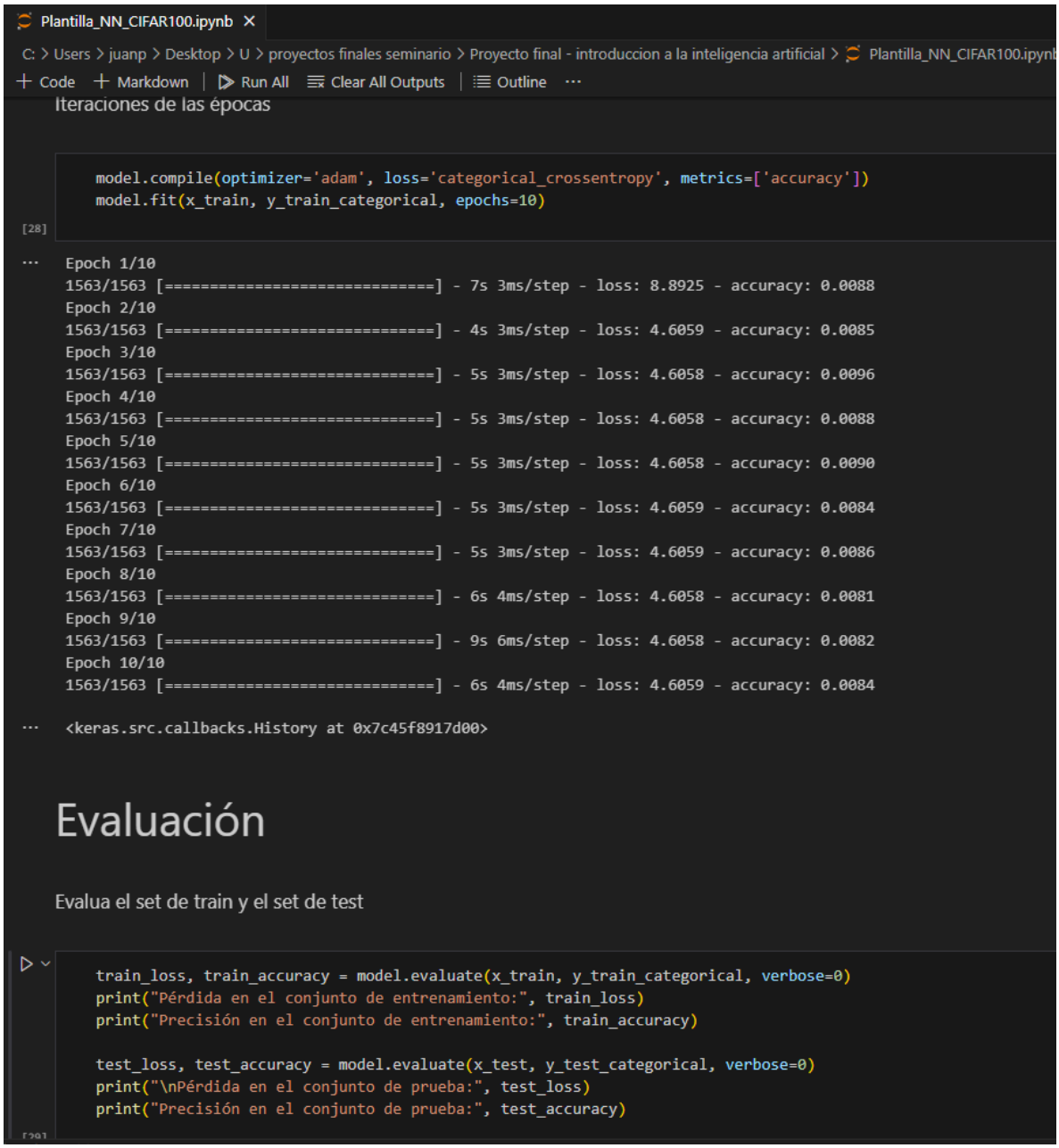
Entrenamiento

Iteraciones de las épocas

```
[28]
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.fit(x_train, y_train_categorical, epochs=10)
```

```
Epoch 1/10
1563/1563 [==============================] - 7s 3ms/step - loss: 8.8925 - accuracy: 0.0088
Epoch 2/10
1563/1563 [==============================] - 4s 3ms/step - loss: 4.6059 - accuracy: 0.0085
Epoch 3/10
1563/1563 [==============================] - 5s 3ms/step - loss: 4.6058 - accuracy: 0.0096
Epoch 4/10
1563/1563 [==============================] - 5s 3ms/step - loss: 4.6058 - accuracy: 0.0088
Epoch 5/10
1563/1563 [==============================] - 5s 3ms/step - loss: 4.6058 - accuracy: 0.0090
Epoch 6/10
```

TRABAJO DE GRADO
Opción Seminario-Diplomado.



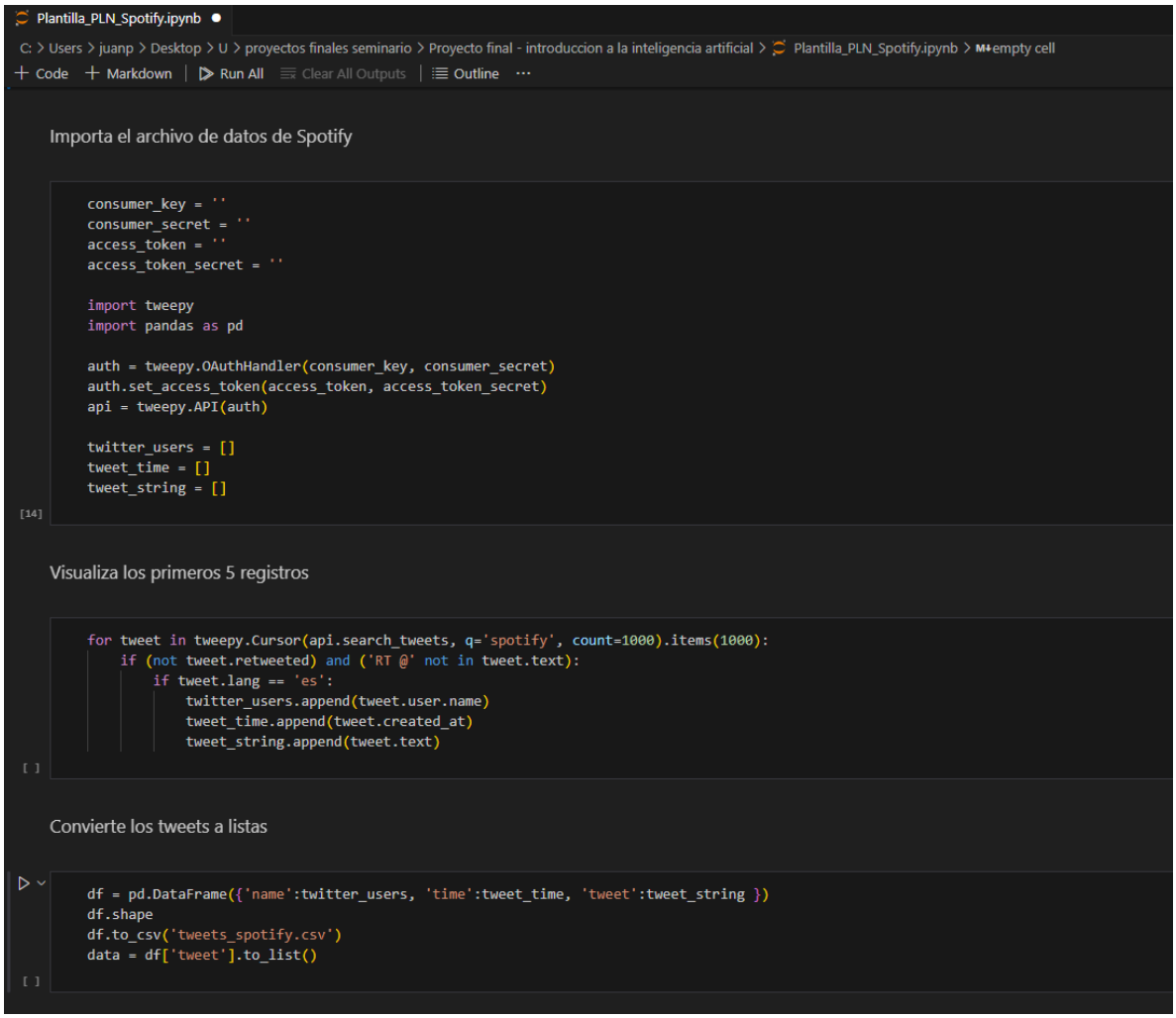
Evaluación

Evalua el set de train y el set de test

```
[29]
    train_loss, train_accuracy = model.evaluate(x_train, y_train_categorical, verbose=0)
    print("Pérdida en el conjunto de entrenamiento:", train_loss)
    print("Precisión en el conjunto de entrenamiento:", train_accuracy)

    test_loss, test_accuracy = model.evaluate(x_test, y_test_categorical, verbose=0)
    print("\nPérdida en el conjunto de prueba:", test_loss)
    print("Precisión en el conjunto de prueba:", test_accuracy)
```

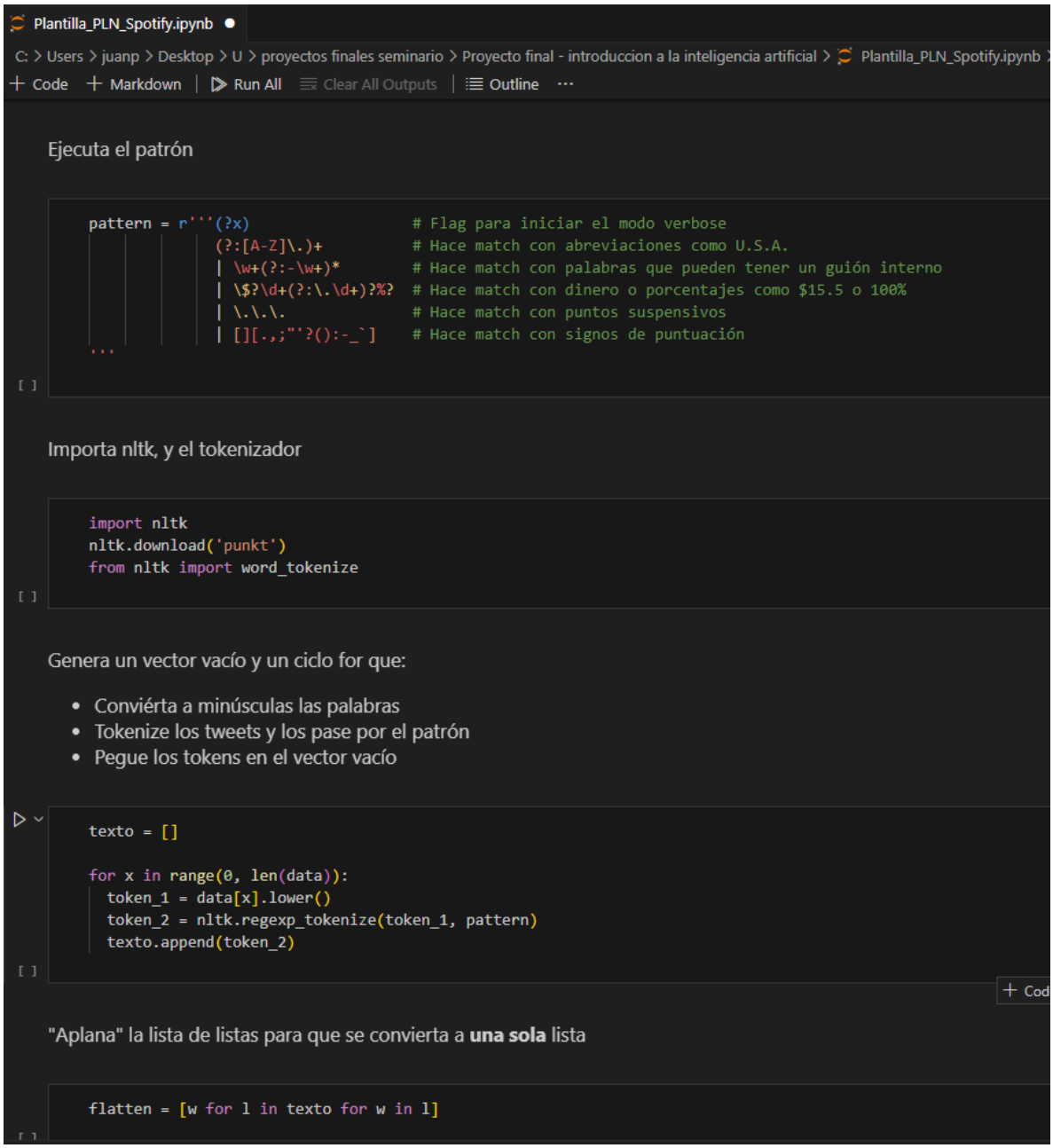
```
Pérdida en el conjunto de entrenamiento: 4.605103969573975
Precisión en el conjunto de entrenamiento: 0.01001999992877245

Pérdida en el conjunto de prueba: 4.605213165283203
Precisión en el conjunto de prueba: 0.009999999776482582
```

Ejercicio 3:


Plantilla_PLN_Spotify.ipynb
C: > Users > juanp > Desktop > U > proyectos finales seminario > Proyecto final - introduccion a la inteligencia artificial > Plantilla_PLN_Spotify.ipynb > empty cell
Code
Markdown
Run All
Clear All Outputs
Outline
Importa el archivo de datos de Spotify
consumer_key = ''
consumer_secret = ''
access_token = ''
access_token_secret = ''
import tweepy
import pandas as pd
auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
api = tweepy.API(auth)
twitter_users = []
tweet_time = []
tweet_string = []
[14]
Visualiza los primeros 5 registros
for tweet in tweepy.Cursor(api.search_tweets, q='spotify', count=1000).items(1000):
if (not tweet.retweeted) and ('RT @' not in tweet.text):
if tweet.lang == 'es':
twitter_users.append(tweet.user.name)
tweet_time.append(tweet.created_at)
tweet_string.append(tweet.text)
[ ]
Convierte los tweets a listas
df = pd.DataFrame({'name':twitter_users, 'time':tweet_time, 'tweet':tweet_string })
df.shape
df.to_csv('tweets_spotify.csv')
data = df['tweet'].to_list()
[ ]

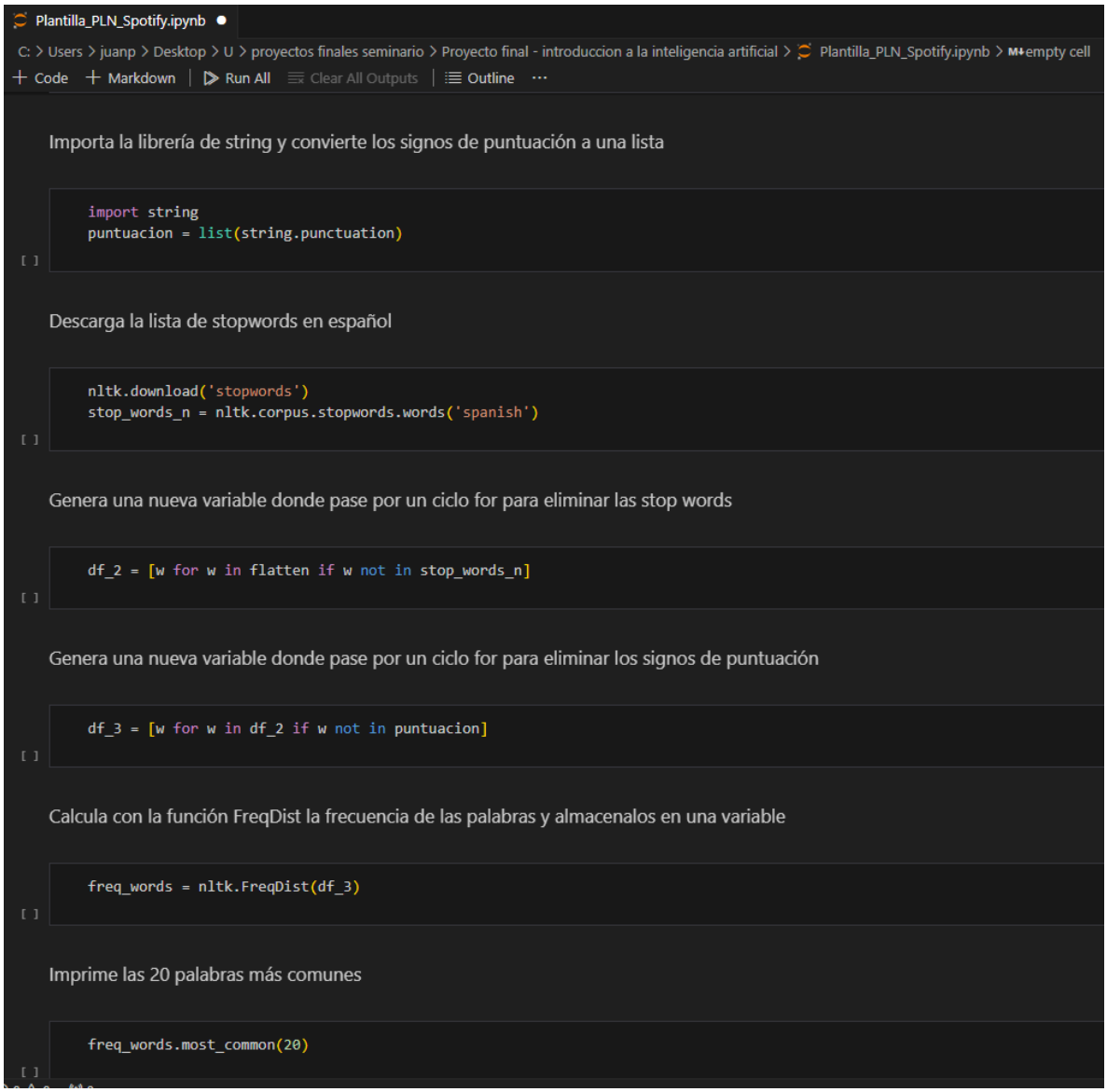
UNIREMINGTON
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

TRABAJO DE GRADO
Opción Seminario-Diplomado.



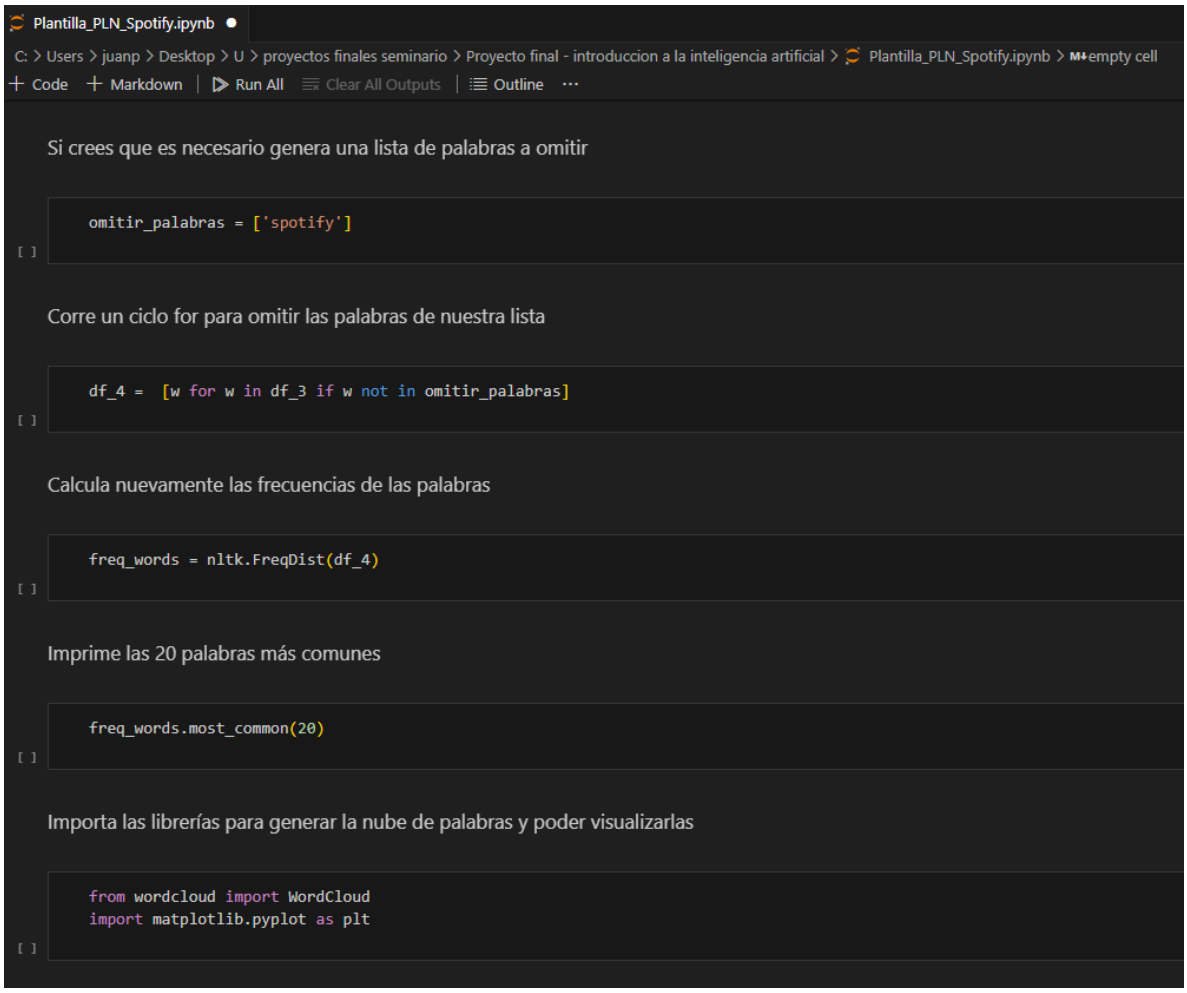
UNIREMINGTON
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

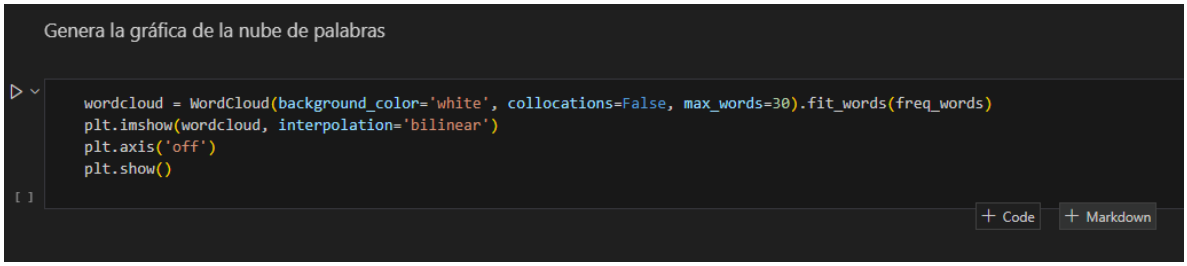
TRABAJO DE GRADO
Opción Seminario-Diplomado.



UNIREMINGTON®
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

TRABAJO DE GRADO
Opción Seminario-Diplomado.





TRABAJO DE GRADO
Opción Seminario-Diplomado.

Proyecto 6: Introducción a machine learning

Avance 1: Contexto y Problema a Resolver

En un mundo cada vez más digitalizado, las empresas buscan optimizar sus ventas a través de redes sociales. El problema radica en la dificultad de gestionar eficientemente las interacciones con los clientes potenciales en estas plataformas. La creación de un asistente virtual de ventas por redes tiene como objetivo mejorar la atención al cliente, la personalización de las interacciones y, en última instancia, aumentar las conversiones de ventas.

Avance 2: Tipo de Predicción y Datos

El tipo de predicción aquí podría centrarse en la probabilidad de conversión de un cliente potencial en función de su interacción con el asistente virtual. Los datos podrían incluir interacciones históricas en redes sociales, como mensajes, respuestas, tiempos de respuesta, preferencias del cliente, historial de compras si está disponible y comportamiento en línea relacionado con la marca.

Avance 3: Roles y Áreas Involucradas

1. **Desarrollador de Machine Learning:** Encargado de crear y entrenar el modelo de machine learning para analizar datos de interacciones y predecir la probabilidad de conversión.
2. **Expertos en Ventas y Marketing:** Proporcionan información sobre el flujo de ventas, las tácticas efectivas en redes sociales y las métricas relevantes para evaluar el rendimiento del asistente virtual.
3. **Ingeniero de Datos:** Responsable de recopilar, limpiar y preparar los datos provenientes de las redes sociales y otros sistemas de la empresa para su uso en el modelo de machine learning.
4. **Especialista en Experiencia del Usuario (UX):** Colabora en el diseño de la interfaz del asistente virtual para garantizar una experiencia fluida y satisfactoria para los usuarios.

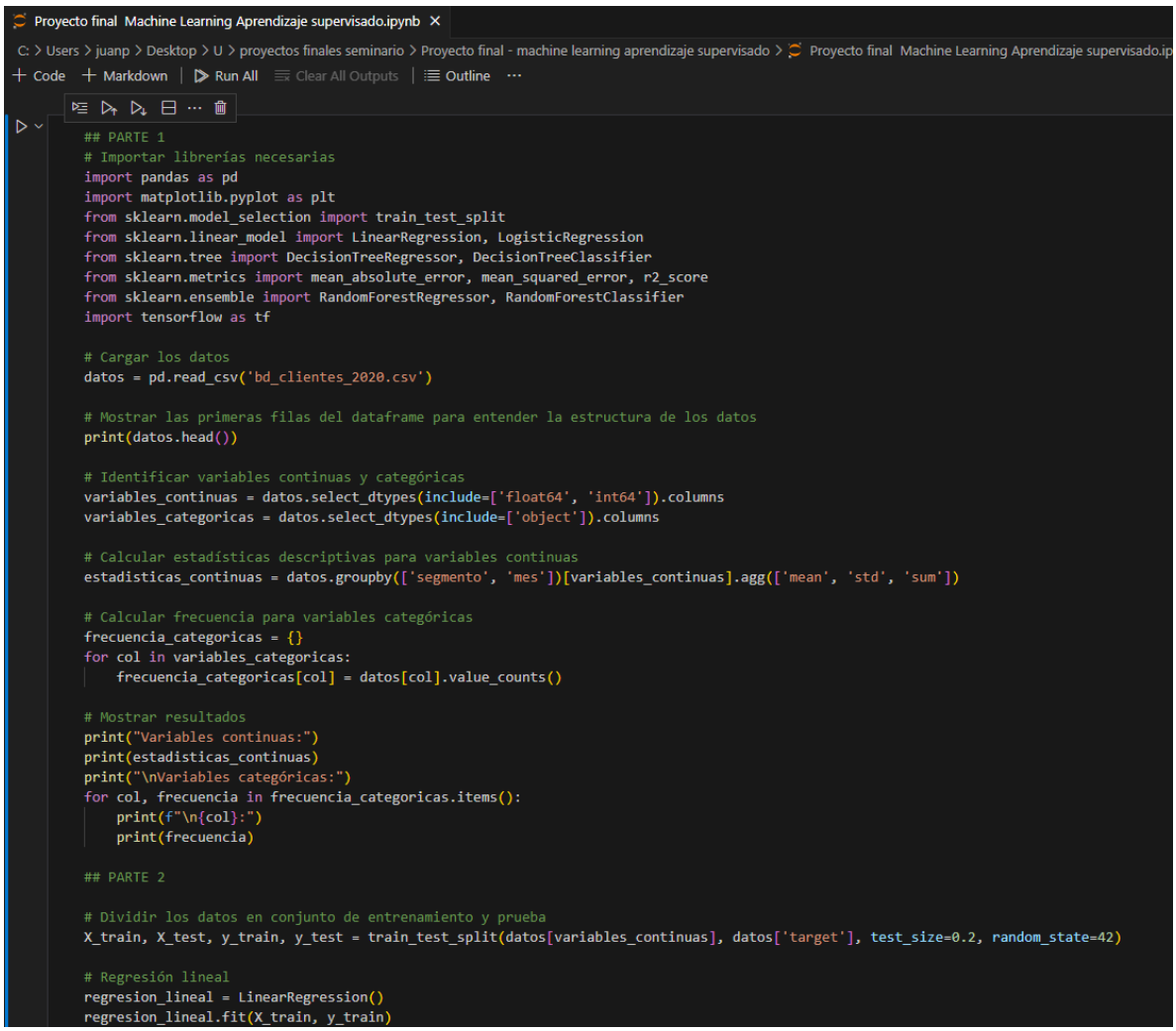
Retos:

- **Calidad de los Datos:** Los datos de interacciones en redes sociales pueden ser ruidosos y variados. Limpiar y estructurar estos datos puede ser desafiante.
- **Desarrollo del Modelo:** Crear un modelo preciso que pueda predecir con confiabilidad la probabilidad de conversión basándose en datos de redes sociales.

- **Interfaz y Experiencia del Usuario:** Diseñar una interfaz intuitiva y atractiva para el asistente virtual que garantice una interacción positiva y efectiva con los clientes potenciales.

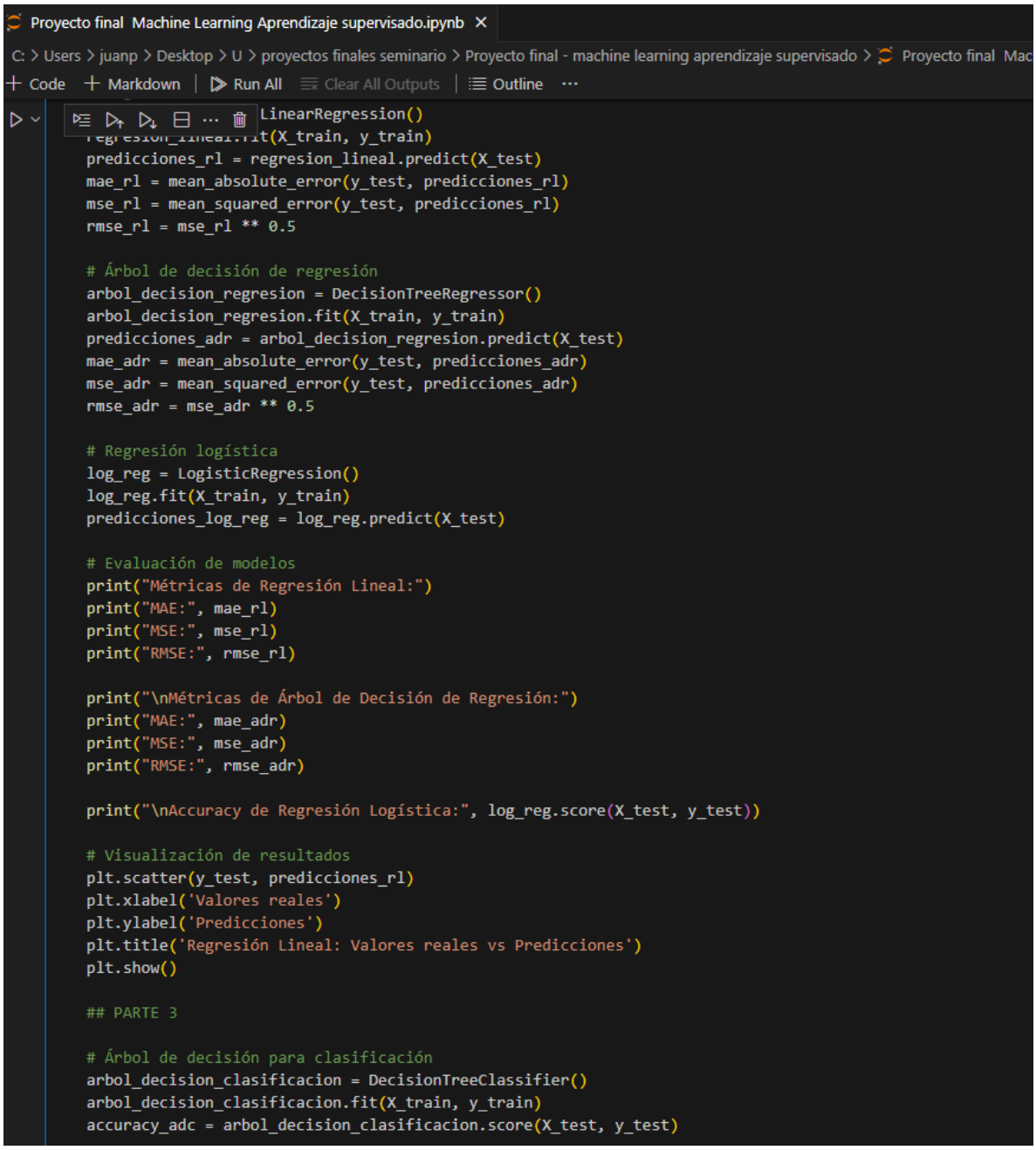
Este proyecto podría agregar un valor significativo al mejorar la eficiencia de las ventas en redes sociales y aumentar la tasa de conversión al ofrecer una interacción más personalizada y efectiva con los clientes.

Proyecto 7: Machine learning aprendizaje supervisado



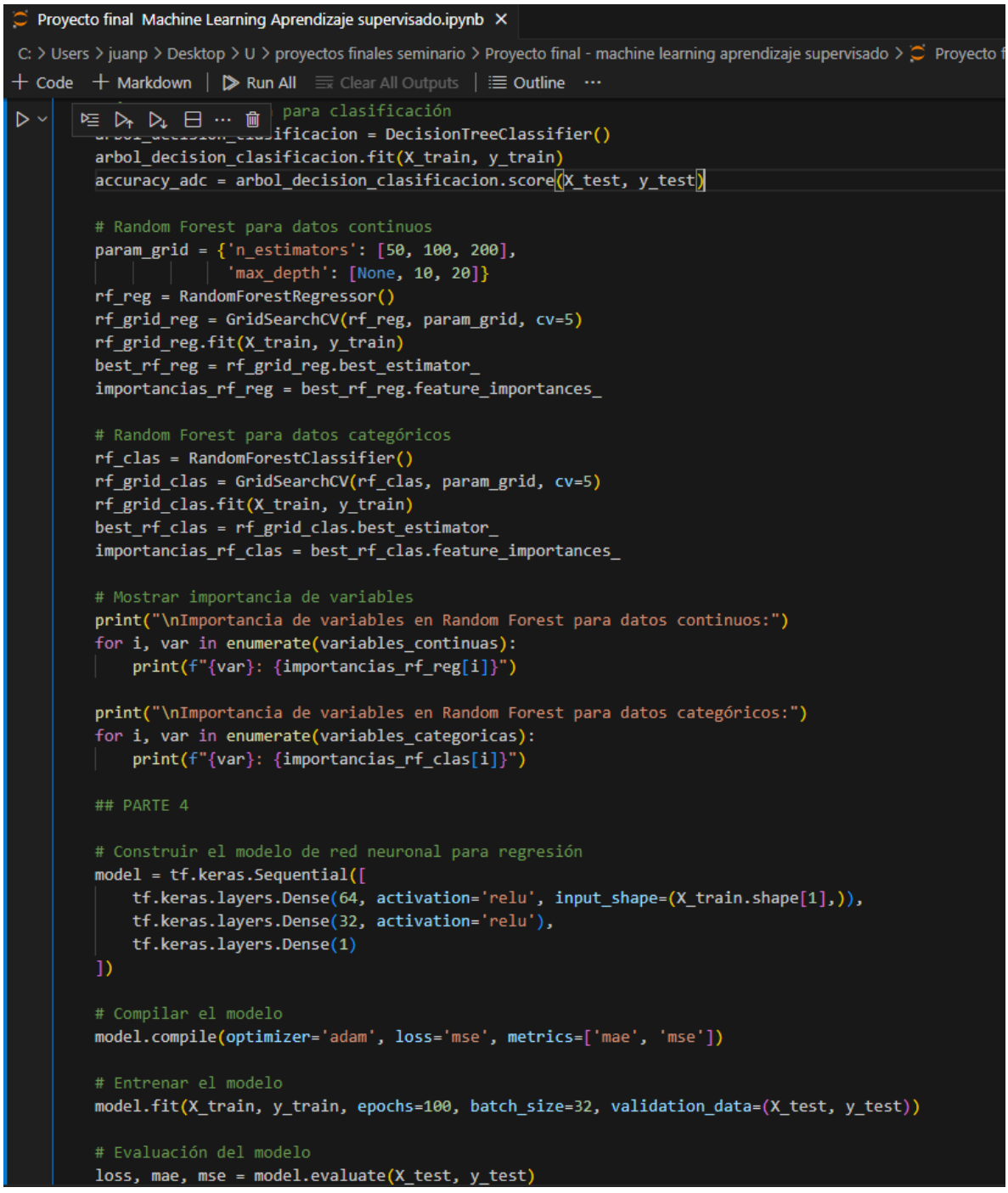
UNIREMINGTON
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

TRABAJO DE GRADO
Opción Seminario-Diplomado.



UNIREMINGTON
CORPORACIÓN UNIVERSITARIA REMINGTON
RES. 2661 MEN JUNIO 21 DE 1996

TRABAJO DE GRADO
Opción Seminario-Diplomado.



```python
# Evaluación del modelo
loss, mae, mse = model.evaluate(X_test, y_test)
print("Loss:", loss)
print("MAE:", mae)
print("MSE:", mse)
```